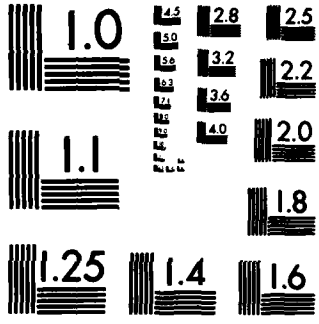
AD-A155 399 FINAL REPORT OF APRIL 1985(U) EIDGENOESSISCHE TECHNISCHE HOCHSCHULE ZURICH (SWITZERLAND) J NIEVERGELT APR 85 DAJA37-82-C-0058 1/1

UNCLASSIFIED F/G 9/2 NL



Final Report of April 1985
ERO contract No. DAJA 37-82-C-0058

Principal investigator: Prof. J. Nievergelt
ETH Zürich, Switzerland

Summary of activities

1) Workshops on Computational Geometry

a) Workshops 1982 -1985

The *financial support of ERO* made it possible to initialize a series of Workshops on Computational Geometry. The first two were held in 1982 and 1983 at ETH in Zurich. Summaries of these workshops have been presented in the Quarterly Progress Reports of June 1982 and March 1983.

Motivated by the success of these two workshops Prof. Nef and Dr. Bieri organized the workshop in 1984 at University of Berne, Switzerland (see Annual Progress Report of June 1984).

This year the workshop was held at University of Karlsruhe, West-Germany. It is planned to continue this Workshop on Computational Geometry as an annual event.

b) Summary of the Workshop held in Karlsruhe on March 28th - 29th, 1985

The following trends were apparent at the Karlsruhe workshop:

- F. Aurenhammer from the Technical University of Graz, Austria, presented *modified Voronoi diagrams* to investigate distance problems. Voronoi diagrams divide a d-dimensional Euclidean space into disjoint regions of identical answers such that all points lying in one region are closer to one of a set of points than to any other point of this set. Modified Voronoi diagrams use points with weights and generalized (non-Euclidean) distance functions.
- O. Fries of Saarland University, Germany, works on methods for determining the area of a planar subdivision in which a certain point lies. The problem lies in the use of dynamic subdivisions, where lines are added or removed dynamically. An algorithm was developed which allows to locate an area in $O(\log^2 n)$ time while only requiring $O(\log^4 n)$ time for insertion or deletion of lines. This compares to an optimal solution for a static subdivision requiring O(log n) location time and O(n log n) preprocessing time.
- K. Mehlhorn and St. Näher of Saarland University are using the method of *fractional cascading* to make complex geometric data structures dynamical.
- W. Schilling of the University of Dortmund, Germany, investigates how external storage techniques can be used in order to reduce the O(n) internal storage needed in *divide-and-conquer* algorithms for solving the rectangle intersection problem.
- R. Klein and O. Nurmi of the University of Karlsruhe presented algorithms for the efficient computation of *direct inclusions* of geometric objects. (Direct inclusion of object A in object B means that A is not included in any object C included in B.) They transform the inclusion problem into the problem of determining the *direct dominance* of points defining the objects.
- H. Bieri and W. Nef of the University of Berne, Switzerland, are investigating properties of the Euler characteristics of polyhedrons and presented an algorithm for its computation.
- H. Hagen of Arizona State University, USA, is using *shading algorithms* based on *ray tracing* to visualize areas of critical curvature of free form surfaces. He hopes that shading will show problem areas on the modelled surfaces, which would not be detectable from the type of graphic image conventionally used in CAD. This is especially true for the *twist problem* which does not have an intuitive meaning.

- S. Abramowski and H. Müller of the University of Karlsruhe are developing efficient algorithms for one-dimensional queries in a three dimensional space. These are required for *ray tracing*. Their algorithms are based on hierarchies of containers of sets of objects. The three-dimensional scene is transformed into a dual space, where the query is solved by determining the element of a partition which contains a certain point.
- T. Spindler of the University of Würzburg investigates the possibilities of using an *array of pixel processors* for image generation. This would allow to draw lines in constant time.
- B. Brüderlin of ETHZ reported on the use of Prolog for the construction of geometric objects which are defined by constraints (see below).
- P. Widmayer of the University of Karlsruhe is using a *branch and bound* technique to solve the layout problem in VLSI design. To this end a parameter driven heuristic was developed which allows to compute approximate solutions using considerably less time than the optimal solution would require.
- K. Hinrichs of ETHZ reported on the implementation of the *grid file system* (see below) and gave an evaluation of its performance. It was shown that with non-uniformly distributed data points the grid file makes good usage of external storage while minimizing the number of physical I/O operations.
- K. Simon of Saarland University is investigating the problem of intersecting spatial objects. He presented an algorithm to compute the intersection of a convex polyhedron and an object with smooth surface. The algorithm reduces the problem to two dimensions using the *hierarchical representation* of a convex polyhedron.
- R. Gnatz and U. Hill-Samelson of the Technical University of Munich, Germany, defined an abstract data type for *Euler operators*. This can be used for the construction of objects in CAD.

2) Own Research

The grant from ERO has been a welcome supplement to other funds that have supported research on *interactive systems* and *algorithms and data structures*, the two major areas of research in our team. We try to create links between these two rather different fields, and we use experience gained and software written for one project in others. Thus the activities supported directly by ERO are embedded in a wider scope, which we now survey briefly. From time to time, these activities have been described in quarterly progress reports to ERO.

1. Grid File Data Management System

The grid file program for storing multidimensional point data and geometric objects [7, 8, 9] is a portable data management package. It has been implemented on VAX/VMS and on several personal computers including Lilith, Smaky PC of EPFL and the M3 multi processor of the Institute for Electronics of ETHZ. On the PCs graphical demonstration programs have been implemented [6], which demonstrate how proximity queries on sets of simple spatial objects are performed with the grid file.

2. Prolog Interpreter

To gain experience with this new programming language and its efficient implementation, an interpreter for Prolog has been written [12]. This interpreter allows calls to Modula procedures from Prolog, and the interpreter can be called from Modula programs. This permits the applications programmer to use Prolog where its strength lies - automated deduction - while using a conventional language for everything else, especially real arithmetic.
Prolog has been used for automatic generation of geometric objects defined by a set of constraints [4]. After these experimental implementations we are now improving this Prolog interpreter. It has been transported to a VAX system and is being used at the Brown Boveri Research Center at Baden-Dättwil for the development of expert systems, as well as a query and manipulation language for the grid file package.

3. Sweep Algorithms for Geometrical Data Processing

Plane sweep algorithms have been theoretically known to be efficient for the analysis of two dimensional geometrical configurations. In his PhD thesis G. Beretta [2] showed that they are efficient and robust in practical applications. It was confirmed that it is possible to write a problem independent kernel, which can be easily adapted to different kinds of applications. Based on the experience gained in this project we

were able to implement a general plane sweep algorithm on the Apple II micro-computer. This proves that plane-sweep algorithms can be implemented efficiently even on small systems.
Unfortunately it became apparent that our hope for an efficient three dimensional generalization of plane sweep would not be fullfilled. An O(n log n) algorithm for the intersection of polyhedra could only be developed with the limitation to convex objects [5]. For practical purposes this is quite a strong limitation. We therefore did not pursue this problem any further and did not implement any "space sweep algorithm".

4. Modula-2 Compiler for Motorola 68000

Our Modula-2 compiler for Motorola 68000 has been implemented on the Smaky-8 and Smaky-100 computers of the Swiss Federal Institute of Technology in Lausanne (EPFL). It is marketed by Epsitec for these machines. Recently the compiler has also been implemented on the Apple Lisa and MacIntosh PCs.

5. Application-Independent Dialog Control of Interactive Systems

The experimental interactive system XS-2 [18, 3], which has been transported last year to Smaky and VAX, has now been implemented on the M3 system. Porting of XS-2 to a different machine is difficult since it uses many low level operating system features. Therefore E. Biagioni developed a simple, easily transportable dialogue front end to interface an interactive application program to an operating system. It supplies the user with general commands based on the sites, modes, trails model of dialogue control.

6. Interactive Educational Demonstration Programms

Our model for the development of interactive educational programs is based on a separation of dialogue control and content. The frame program (a network of dynamic pages and corresponding dialogue control commands) is generated automatically by a frame program generator. This approach is described in a text book [15] which also includes examples for teaching purposes.

Publications

[1] D. Ackermann, J. Nievergelt, *Die Fünffinger Maus: Eine Fallstudie zur Synthese von Hardware und Physiologie, Software und Psychologie*, submitted.

[2] G. Beretta, *An implementation of a plane sweep algorithm on a personal computer*, Diss. ETH 7538, 1984.

[3] E. Biagioni, J. Nievergelt, H. Sugaya, J. Stelovsky, *Can an operating system support consistent user dialogs? Experience with the prototype XS-2*, submitted.

[4] B. Brüderlin, *Using Prolog for the construction of geometric objects defined by constraints*, to appear, Proc Eurocal 85, Linz.

[5] S. Hertel, M. Mantyla, K. Mehlhorn, J. Nievergelt, *Space-sweep solves intersection of convex polyhedra*, Acta Informatica 21, 501-519, Nov 1984.

[6] K. Hinrichs and J. Nievergelt, *The grid file: A data structure designed to support proximity queries on spatial objects*, OUTPUT Nr. 5, Mai 1984, 51-578, Fachpresse Goldach.

[7] K. Hinrichs, *Implementation of the grid file: design concepts and experience*, to appear in BIT 1985.

[8] K. Hinrichs, *The grid file system: implementation and case studies of applications*, Diss. ETH No. 7734, 1985.

[9] H. Hinterberger, *A Pascal grid file program and an experiment in concurrent access control*, Diplomarbeit Informatik ETH, 1984.

[10] H. Hinterberger, *Interacting with Videotex*, OUTPUT Nr. 3, 53-57, and Nr. 8, 39-42, 1984, Fachpresse Goldach.

[11] A. Kierulf, J. Nievergelt, *Computer Go: A smart board and its applications*, submitted.

[12] C. Muller, *A Prolog front end to the grid file*, Diplomarbeit Informatik ETH, 1984.

[13] J. Nievergelt, H. Hinterberger and K. C. Sevcik: *The Grid File: an adaptable, symmetric multikey file structure, ACM Trans. Database Systems*, 9, 1, 38-71, Mar 1984.

[14] J. Nievergelt, *Die n-te Generation*, GI Spektrum 7, 237-242, Nov 1984.

[15] J. Nievergelt, A. Ventura and H. Hinterberger, *Interactive Programs for Education: Philosophy, Techniques, and Examples*, Addison Wesley, to appear 1985.

[16] J. Nievergelt, *Issues in the design of human-computer interfaces*, Proc. NATO Advanced Study Institute on Pictorial Information Systems in Medicine, Springer Verlag, to appear 1985.

[17] J. Nievergelt, K. Hinrichs, *Storage and access structures for geometric data bases*, Intern. Conf. on Foundations of Data Structures, Kyoto, 1985.

[18] J. Stelovsky, XS-2: *The user interface of an interactive system*, Diss. ETH No. 7425, 1984.

3) Software Packages

Easy

EASY provides a standardized, application-independent user interface for highly interactive application programs. The application program retains full control and invokes EASY procedures rather than the usual input routines. EASY provides for command definition and invocation as well as for data structuring (hierarchical file system).

EASY is written in Modula-2 and runs on the personal computer Lilith, on which it was developed, and on DEC VAX using HP 2648 and Zenith terminals. It is usable with alphanumeric terminals with randomly-positioned writing, and can take advantage of graphic terminals if they are available. The main part of the software is machine-independent, and therefore easily portable; the machine-dependent part is concentrated in a small number of modules that define a virtual machine. These modules are general-purpose and are used without modification also by Prolog and by the Grid file system.

One application of EASY has been as a user interface to the Prolog interpreter described below.

Prolog

"Modula--Prolog" is a software package written in Modula-2, offering tools for constructing Prolog interpreters which can interact in many ways with other Modula-2 programs. The package is designed as a Modula-2 library module, which defines and implements the interface between Prolog and Modula-2. In *"Modula--Prolog"*, the basic Prolog functions (parse, prove, unparse, unification, ...) are isolated and can be called separately as library procedures from various Modula-2 programs. Input and output operations are user-controllable, i.e. *"Modula--Prolog"* fits easily in existing dialog structures, and may be adapted to a variety of screen- and window-packages.

"*Modula--Prolog*" can be used by Modula-2 applications as a problem solving tool in the background. When there is need for a deductive component, the Modula-2 application may call Prolog to execute some specific job (Prolog query) and afterwards continue with the results obtained from Prolog. Information exchange between Prolog and Modula-2 is based on Prolog terms. A term in "*Modula--Prolog*" is a coded (hidden) representation of a term as defined in the Prolog language (as defined in Clocksin & Mellish: Programming in Prolog, Springer 1981). "*Modula--Prolog*" provides two sets of procedures for processing terms: *term-assembling procedures* and *term-disassembling procedures*, which allow, on the one hand, the construction of a term out of its basic components (atoms, functors, numbers, ...) and, on the other hand, the separation of the basic components of a term for further use in Modula-2.

"Modula--Prolog" provides features for easily extending the set of *Prolog builtin predicates*, i. e. it is an

instrument for building powerful user-tailored Prolog systems (e.g. integration of graphics or database operations in the Prolog language). These user-defined builtin predicates can manipulate Prolog terms in their full generality, again by using the term-assembling and term-disassembling procedures. The writer of a builtin predicate can use the Prolog unification procedure to perform variable bindings and consistency checks.

Grid file system

The grid file is a data struture designed to handle large amounts of multidimensional point data in an efficient way. A grid file adapts its shape dynamically to its contents under insertions and deletions in order to guarantee retrieval of a single record in two disk accesses. All keys (attributes) are treated symmetrically (the grid file does not favor a primary key at the expense of secondary keys), thus queries that involve different attributes are processed with equal efficiency. The grid file was designed to support efficient proximity queries, such as range and nearest neighbor queries on multidimensional data.

The grid file system is a complete portable software package for storing multidimensional point data on secondary storage which is based on the grid file concept. It provides procedures for performing the following operations on grid files:

- creating, deleting, opening and closing grid files;
- inserting and deleting records (which may be identified by keys of different types) in a grid file;
- changing non-key information in a record;
- point queries, range queries, user-defined queries and join queries.

The grid file system is written in Modula-2 and runs on DEC-VAX under the VMS-operating system. Furthermore there exist Modula-2 implementations on different personal computers.

The logic programming language Prolog is especially well suited as a query language for data bases. The natural correspondence between Prolog facts and grid files or relations in data bases and the deduction facilities provided by Prolog allow the advanced use of data. A Prolog front end to the grid file system allows the user to work interactively with the data stored in grid files and makes available the full power of Prolog.

END

FILMED

7-85

DTIC